A logical analysis of entanglement and separability in quantum higher-order functions

F. Prost and C. Zerrari

LIG
46, av Félix Viallet, F-38031 Grenoble, France
Frederic.Prost@imag.fr

Abstract. We present a logical separability analysis for a functional quantum computation language. This logic is inspired by previous works on logical analysis of aliasing for imperative functional programs. Both analyses share similarities notably because they are highly non-compositional. Quantum setting is harder to deal with since it introduces non determinism and thus considerably modifies semantics and validity of logical assertions. This logic is the first proposal of entanglement/separability analysis dealing with a functional quantum programming language with higher-order functions.

1 Introduction

The aim of high level programming language is to provide a sufficiently high level of abstraction in order both to avoid unnecessary burden coming from technical details and to provide useful mental guidelines for the programmer. Quantum computation [4] is still in its prime and quantum programing languages remain in need for such abstractions. Functional quantum programing languages have been proposed and offer ways to handle the no-cloning axiom via linear λ-calculi [9, 7]. In [1] is developed QML in which a purely quantum control expression is introduced in order to represent quantum superposition in programming terms. Another crucial ingredient of quantum computation is the handling of entanglement of quantum states during computation. Indeed without entanglement it is possible to efficiently simulate quantum computations on a classical computer [10]. A first step to deal with entanglement, and its dual: separability, has been done in [5] in which a type system is provided in order to approximate the entanglement relation of an array of quantum bits.

Quantum bits entanglement analysis shares some similarities with variables name aliasing analysis.Indeed, aliasing analyzes are complicated since an action on a variable of a given name may have repercussions on another variable having a different name. The same kind of problems occur between two entangled quantum bits : if one quantum bit is measured then the other one can be affected. In both cases there is a compositionality issue: it is hard to state anything about a program without any knowledge of its context. It seems therefore sensible to try to adapt known aliasing analysis techniques to the quantum setting.

In this paper we follow the idea developed in [2] and adapt it for entanglement/separability analysis in a functional quantum programing language with higher order functions. The work of [2] has to be adapted in a non deterministic setting, which is inherent of quantum computation, making the semantics and soundness of the logic radically different. Moreover, our results are a strict improvement over [5] in which only first order functions are considered.

1.1 outline of the paper

We first start by giving the definition of the dual problems of entanglement and separability, together with quick reminders on quantum computation, in section 2. Then, in section 3, we present a functional quantum computation language in section for which we define an entanglement logic in section 4. Finally, we conclude in section 5.

2 Separability and Entanglement

A n qubits register is represented by a normalized vector in a Hilbert 2^n-dimension space that is the tensorial product of n dimension 2 Hilbert spaces on $\mathbb{C}^2$. Each 2 dimension subspace represents a qubit. For a given vector, written $|\varphi\rangle$, qubits can be either entangled or separable.

Definition 1 (Entanglement, Separability). *Consider $|\varphi\rangle$ a n qubits register. φ is separable if it is possible to partition the n qubits in two non empty sets A, B, two states $|\varphi_A\rangle$ and $|\varphi_B\rangle$ describing A and B qubits, such that $|\varphi\rangle = |\varphi_A\rangle \otimes |\varphi_B\rangle$, where $|\varphi_A\rangle$ and $|\varphi_B\rangle$,otherwise it is said entangled.*

By extension, two qubits q, q' are separable if and only if there exists a partition A, B, two states $|\varphi_A\rangle$ and $|\varphi_B\rangle$ describing A and B qubits, such that $|\varphi\rangle = |\varphi_A\rangle \otimes |\varphi_B\rangle$, with $q \in A$ and $q' \in B$. Otherwise q, q' are entangled.

Definition 2 (Entanglement relation). *Let a n qubits register be represented by $|\varphi\rangle$. The entanglement relation of $|\varphi\rangle$, $\mathbf{E}(|\varphi\rangle)$, over qubits of the register is defined as follows: $(x, y) \in \mathbf{E}(|\varphi\rangle)$ if and only if x and y are entangled.*

The entanglement relation is an equivalence relation. It is indeed obviously symmetric and reflexive. It is transitive because if $(x, y) \in \mathbf{E}(|\varphi\rangle)$ and $(y, z) \in \mathbf{E}(|\varphi\rangle)$. It is possible to find a partition X, Z (with $x \in X$ and $z \in Z$) and $|\varphi_X\rangle, |\varphi_Z\rangle$ such that $|\varphi\rangle = |\varphi_X\rangle \otimes |\varphi_Z\rangle$. y is either in X or Y then either (x, y) or (y, z) is not in $\mathbf{E}(|\varphi\rangle)$, thus the result by contradiction.

3 $\boldsymbol{\lambda_L^Q}$ a functional quantum computing language

We use a variant of Selinger and Valiron's λ-calculus [7] as programming language. Instead of considering arbitrary unitary transformations we only consider three: quantum phase $\mathfrak{T}$, Hadamard transformation $\mathfrak{H}$, and conditional not $\mathfrak{Cnot}$.

This restriction doest not make our language less general since it forms a universal quantum gates set, see [4]. It makes entanglement analysis simpler. Indeed, only $\mathfrak{Cnot}$ may create entanglement. We also introduce another simplifications: since the calculus may be linear only for quantum bits we do not use all the linear artillery (bang, linear implications etc) but only check that abstractions over quantum bits are linear. Moreover we suppose a fixed number of quantum bits, therefore there are no new operators creating new quantum bits during computation. Indeed as shown in [6] name generation creates nontrivial problems.

Therefore, in the following we suppose the number of quantum bit registers fixed although non specified and refer to it as n.

3.1 Syntax and types

Definition 3 (Terms and types). *λ_L^Q terms and types are inductively defined by:*

$$\begin{aligned} M, N, P ::= {} & x \mid q_i \mid \mathbf{1} \mid \mathbf{0} \mid \lambda x : M.N \mid (M \quad N) \\ & \mid \mathbf{1} \mid \mathbf{0} \mid \langle M, N \rangle \mid \pi_i N \mid \mathsf{if}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P \mid \\ & \mathsf{meas} \mid \mathfrak{Cnot} \mid \mathfrak{H} \mid \mathfrak{T} \end{aligned}$$

$$\sigma, \tau ::= \mathbf{B} \mid \mathbf{B}^\circ \mid \sigma \to \tau \mid \sigma \otimes \tau$$

where x denotes names of element of a countable set of variables. q_i, where $i \in \{1..n\}$ are constant names that are used as reference for a concrete quantum bit array. $\mathbf{1}, \mathbf{0}$ are standard boolean constant. $\pi_i N$ with $i \in \{1, 2\}$ is the projection operator. Terms of the third line are quantum primitives respectively for measure, quantum bit initialization and the three quantum gates Conditional not, Hadamard and phase.

We only have two base types $\mathbf{B}$ for bits and $\mathbf{B}^\circ$ for quantum bits, arrow and product types are standard ones.

Note that if quantum bits are constants, there can be quantum bits variable in this λ_L^Q. Consider for instance the following piece of code: $(\lambda x : M.\ \mathsf{if}\ M\ \mathsf{then}\ q_1\ \mathsf{else}\ q_2)$. After reduction x may eventually become either q_1 or q_2. We write q without subscript to denote quantum bit variables.

Definition 4 (Context and typing judgments). *Contexts are inductively defined by:*

$$\Gamma ::= \cdot \mid \Gamma, x : \sigma$$

where σ is not $\mathbf{B}^\circ$.

We define lists of quantum bits variable by:

$$\Lambda ::= \cdot \mid \Lambda, q$$

Typing judgments are of the form:

$$\Gamma; \Lambda \vdash M : \sigma$$

and shall be read as : under the typing context Γ, list of quantum bits variable Λ , the term M is well formed of type σ.

As usual we require that typing contexts and lists are unambiguous. It means that when we write $\Gamma, x : \sigma$ (resp. Λ, q) x (resp. q) is implicitly supposed not to appear in Γ (resp. Λ). Similarly when we write Γ_1, Γ_2 (resp. Λ_1, Λ_2) we intend that Γ_1 and Γ_2 (resp. Λ_1 and Λ_2) are disjoint contexts.

Typing rules are the following :

$$\frac{}{\Gamma; \Lambda \vdash q_i : \mathbf{B}^\circ}[AxQ]$$

$$\frac{}{\Gamma; q \vdash q : \mathbf{B}^\circ}[VarQ] \qquad \frac{}{\Gamma, x : \sigma; \cdot \vdash x : \sigma}[Var]$$

$$\frac{}{\Gamma; \cdot \vdash \mathbf{1} : \mathbf{B}}[AxT] \qquad \frac{}{\Gamma; \cdot \vdash \mathbf{0} : \mathbf{B}}[AxF]$$

$$\frac{\Gamma; \cdot \vdash M : \sigma}{\Gamma, x : \tau; \cdot \vdash M : \sigma}[Wkg]$$

$$\frac{\Gamma, x : \sigma; \Lambda \vdash M : \tau}{\Gamma; \Lambda \vdash \lambda x : \sigma.M : \sigma \to \tau}[\to I] \qquad \frac{\Gamma; \Lambda \vdash M : \sigma \to \tau \quad \Gamma; \cdot \vdash N : \sigma}{\Gamma; \Lambda \vdash (M \quad N) : \tau}[\to E]$$

$$\frac{\Gamma; \Lambda, q \vdash M : \tau}{\Gamma; \Lambda \vdash \lambda q : \mathbf{B}^\circ.M : \mathbf{B}^\circ \to \tau}[\to^\circ I] \quad \frac{\Gamma; \Lambda_1 \vdash M : \sigma \to \tau \quad \Gamma; \Lambda_2 \vdash N : \sigma}{\Gamma; \Lambda_1, \Lambda_2 \vdash (M \quad N) : \tau}[\to^\circ E]$$

$$\frac{\Gamma; \Lambda_1 \vdash M : \tau \quad \Gamma; \Lambda_2 \vdash N : \sigma}{\Gamma; \Lambda_1, \Lambda_2 \vdash \langle M, N \rangle : \tau \otimes \sigma}[\otimes I]$$

$$\frac{\Gamma; \Lambda_1 \vdash M : \mathbf{B} \quad \Gamma; \Lambda_2 \vdash N : \tau \quad \Gamma; \Lambda_2 \vdash P : \tau}{\Gamma; \Lambda_1, \Lambda_2 \vdash \textsf{if } M \textsf{ then } N \textsf{ else } P : \tau}[IF\ I]$$

$$\frac{\Gamma'; \Lambda \vdash M : \tau_1 \otimes \tau_2}{\Gamma; \Lambda \vdash \pi_i M : \tau_i}[\otimes E] ii \in \{1, 2\}$$

$$\frac{\Gamma; \Lambda \vdash M : \mathbf{B}^\circ}{\Gamma; \Lambda \vdash \mathfrak{H}\ M : \mathbf{B}^\circ}[HAD]$$

$$\frac{\Gamma; \Lambda \vdash M : \mathbf{B}^\circ}{\Gamma; \Lambda \vdash \mathfrak{T}\ M : \mathbf{B}^\circ}[PHA]$$

$$\frac{\Gamma; \Lambda \vdash M : \mathbf{B}^\circ}{\Gamma; \Lambda \vdash \textsf{meas } M : \mathbf{B}}[MEAS]$$

$$\frac{\Gamma; \Lambda \vdash M : \mathbf{B}^\circ \otimes \mathbf{B}^\circ}{\Gamma; \Lambda \vdash \mathfrak{Cnot}\ M : \mathbf{B}^\circ \otimes \mathbf{B}^\circ}[CNOT]$$

Where in rule $[\otimes E]$ $\Gamma' = \Gamma, x : \sigma, y : \tau$ if σ and τ are not $\mathbf{B}^\circ$, $\Gamma' = \Gamma, x : \sigma$ (resp. $\Gamma, y : \tau$) if τ (resp. σ) is $\mathbf{B}^\circ$ and σ (resp. τ) is not $\mathbf{B}^\circ$. Λ_2' is build in a symmetrical way, thus Λ_2' is Λ_2 augmented with variables x or y if and only if their type is $\mathbf{B}^\circ$.

λ_L^Q is a standard simply typed λ-calculus with two base types which is linear for terms of type $\mathbf{B}^\circ$. Thus we ensure the no-cloning property of quantum physics (e.g. [4]).

3.2 Operational semantics

Quantum particularities have strong implications in the design of a quantum programming language. First, since quantum bits may be entangled together it is not possible to textually represent individual quantum bits as a normalized vector of $\mathbb{C}^2$. We use $|\mathbf{1}\rangle$ and $|\mathbf{0}\rangle$ as base. Therefore, a quantum program manipulating n quantum bits is represented as a quantum state of a Hilbert space $\mathbb{C}^{2n}$ and constants of type $\mathbf{B}^\circ$ are pointers to this quantum state. Moreover, quantum operators modify this state introducing imperative actions. As a consequence an evaluation order has to be set in order to keep some kind of confluence. Moreover, λ_L^Q reductions are probabilistic. Indeed, quantum mechanics properties induce an inherent probabilistic behavior when measuring the state of a quantum bit.

Definition 5 (λ_L^Q state). *Let $\Gamma; \Lambda \vdash M : \sigma$. A λ_L^Q state is a couple $[|\varphi\rangle, M]$.where $|\varphi\rangle$ is a normalized vector of $\mathbb{C}^{2n}$ Hilbert space and M a λ_L^Q term.*

An example of λ_L^Q state of size $n = 2$ is the following:

$$[|\varphi\rangle, (\lambda q : \mathbf{B}^\circ.\mathsf{if}\ (\mathsf{meas}\ \ q_1)\ \mathsf{then}\ \mathbf{1}\ \mathsf{else}\ (\mathsf{meas}\ \ (\mathfrak{T}\ \ q))\ \ q_2)]$$

where $|\varphi\rangle = \frac{1}{\sqrt{2}}(|\mathbf{0}\rangle + |\mathbf{1}\rangle) \otimes (\frac{2}{3}|\mathbf{0}\rangle + \frac{\sqrt{5}}{3}|\mathbf{1}\rangle)$ q_1 is the quantum bit denoted by $\frac{1}{\sqrt{2}}(|\mathbf{0}\rangle + |\mathbf{1}\rangle)$ and q_2 the one represented by $\frac{2}{3}|\mathbf{0}\rangle + \frac{\sqrt{5}}{3}|\mathbf{1}\rangle$.

We consider call by value reduction rules. Values are defined as usual.

Definition 6 (Values). *Values of λ_L^Q are inductively defined by:*

$$U, V ::= x \mid \mathbf{1} \mid \mathbf{0} \mid q_i \mid \lambda x : \sigma.M \mid \langle V, V\rangle \mid (F\ \ x)$$

Where F is one of the following operators $\pi_i, \mathfrak{Cnot}, \mathfrak{T}, \mathfrak{H}, \mathsf{meas}$

We can now define probabilistic reduction rules. We only mention probabilities to be accurate although we are not going to investigate any related problems in this paper (we do not consider confluence problems etc.).

Definition 7 (Quantum reductions). *We define a probabilistic reduction between λ_L^Q states as:*

$$[|\varphi\rangle, M] \to_p [|\varphi'\rangle, M']$$

That has to be red $[|\varphi\rangle, M]$ reduces to $[|\varphi'\rangle, M']$ with probability p.

Reduction rules are the following:

$$\frac{}{[|\varphi\rangle, (\lambda x:\sigma.M \quad V)] \to_1 [|\varphi\rangle, M\{x := V\}]}[\beta V]$$

$$\frac{[|\varphi\rangle, N] \to_p [|\varphi'\rangle, N']}{[|\varphi\rangle, (M \quad N)] \to_p [|\varphi'\rangle, (M \quad N')]}[\beta]$$

$$\frac{[|\varphi\rangle, N] \to_p [|\varphi'\rangle, N']}{[|\varphi\rangle, (M \quad N)] \to_p [|\varphi'\rangle, (M \quad N')]}[App]$$

$$\frac{[|\varphi\rangle, M] \to_p [|\varphi'\rangle, M']}{[|\varphi\rangle, (M \quad V)] \to_p [|\varphi'\rangle, (M' \quad V)]}[Apc]$$

$$\frac{}{[|\varphi\rangle, \textsf{if } \mathbf{1} \textsf{ then } M \textsf{ else } N] \to_1 [|\varphi\rangle, M]}[IF/T]$$

$$\frac{[|\varphi\rangle, P] \to_p [|\varphi'\rangle, P']}{[|\varphi\rangle, \textsf{if } P \textsf{ then } M \textsf{ else } N] \to_p [|\varphi\rangle, \textsf{if } P' \textsf{ then } M \textsf{ else } N]}[IF]$$

$$\frac{}{[|\varphi\rangle, \textsf{if } \mathbf{0} \textsf{ then } M \textsf{ else } N] \to_1 [|\varphi\rangle, N]}[IF/F]$$

$$\frac{i \in \{1, 2\}}{[|\varphi\rangle, \pi_i\langle V_1, V_2\rangle] \to_1 [|\varphi\rangle, V_i]}[\pi]i$$

$$\frac{[|\varphi\rangle, M] \to_p [|\varphi'\rangle, M']}{[|\varphi\rangle, \langle M, N\rangle] \to_p [|\varphi'\rangle, \langle M', N\rangle]}[LFT] \qquad \frac{[|\varphi\rangle, N] \to_p [|\varphi'\rangle, N']}{[|\varphi\rangle, \langle V, N\rangle] \to_p [|\varphi'\rangle, \langle V, N'\rangle]}[RGT]$$

$$\frac{}{[|\varphi\rangle, (\mathfrak{T} \quad q_i)] \to_1 [\mathfrak{T}^i(|\varphi\rangle), q_i]}[PHS] \qquad \frac{}{[|\varphi\rangle, (\mathfrak{H} \quad q_i)] \to_1 [\mathfrak{H}^i(|\varphi\rangle), q_i]}[HDR]$$

$$\frac{}{[\alpha|\varphi_{\mathbf{0}}\rangle + \beta|\varphi_{\mathbf{1}}\rangle, (\textsf{meas} \quad q_i)] \to_{|\alpha|^2} [|\varphi_{\mathbf{0}}\rangle, \mathbf{1}]}[MEF]$$

$$\frac{}{[\alpha|\varphi_{\mathbf{0}}\rangle + \beta|\varphi_{\mathbf{1}}\rangle, (\textsf{meas} \quad q_i)] \to_{|\beta|^2} [|\varphi_{\mathbf{1}}\rangle, \mathbf{0}]}[MET]$$

$$\frac{}{[|\varphi\rangle, (\mathfrak{Cnot} \quad \langle q_i\rangle q_j)] \to_1 [\mathfrak{Cnot}^{i,j}(|\varphi\rangle), \langle q_i, q_j\rangle]}[CNO]$$

In rules $[MET]$ and $[MEF]$, let $|\varphi\rangle = \alpha|\varphi_{\mathbf{0}}\rangle + \beta|\varphi_{\mathbf{1}}\rangle$ be normalized with

$$|\varphi_{\mathbf{1}}\rangle = \textstyle\sum_i = 1^n \alpha_i |\phi_i^{\mathbf{1}}\rangle \otimes |\mathbf{1}\rangle \otimes |\psi_i^{\mathbf{1}}\rangle$$
$$|\varphi_{\mathbf{0}}\rangle = \textstyle\sum_i = 1^n \beta_i |\phi_i^{\mathbf{0}}\rangle \otimes |\mathbf{0}\rangle \otimes |\psi_i^{\mathbf{0}}\rangle$$

. where $|\mathbf{1}\rangle$ and $|\mathbf{0}\rangle$ is the ith quantum bit.

We say that the set of rules containing $[\beta]$, $[\beta V]$, $[App]$, $[Apc]$, $[Apv]$, $[IF]$, $[IF/F]$, $[IF/T]$, $[\pi]i$, $[LFT]$, $[RGT]$ is the purely functional part of λ_L^Q, the other rules are the quantum part of λ_L^Q.

Based on this reduction rules one can define reachable states, by considering the reflexive-transitive closure of $\rightarrow_p$. One has to compose probabilities along a reduction path. Therefore $[|\varphi'\rangle, M']$ is reachable from $[|\varphi'\rangle, M']$, if there is a non zero probability path between those states. More precisions can be found in [7].

Computations of a λ_L^Q term are done from an initial state where all registers are set to $|\mathbf{0}\rangle$: $|\varphi_{\mathbf{0}}\rangle = |\mathbf{0}\rangle \overbrace{\otimes \ldots \otimes}^{n-1} |\mathbf{0}\rangle$

Proposition 1 (Subject Reduction). *Let $\Gamma, \Lambda \vdash M : \tau$ and $M \rightarrow_p M'$, then $\Gamma, \Lambda \vdash M' : \tau$*

Proof. From the typing point of view λ_L^Q is nothing more than a simply typed λ-calculus with constants for quantum bits manipulations. Note that $\mathfrak{T}, \mathfrak{H}, \mathfrak{Cnot}$ act as identity functions (from the strict λ-calculus point of view). The measurement is simple to deal with since it only returns constant (hence typable in any contexts).

4 Entanglement logic for λ_L^Q

We present a static analysis for the study of the entanglement relation during a quantum computation. The idea that we follow in this paper is to adapt the work [2] to the quantum setting. The logic is in the style of Hoare [3] and leads to the following notation:

$$\{C\}M :^{\Gamma;\Lambda,} u\{C'\}$$

where C is a precondition, C' is a post-condition, M is the subject and u is its anchor (the name used in C' to denote M value). Informally, this judgment can be red: if C is satisfied, then after the evaluation of M, whose value is denoted by u in C', C' is satisfied. $\Gamma; \Lambda$ is the typing context of M and Δ is the anchor typing context : it is used in order to type anchors within assertions. Indeed, anchors denote terms and have to be typed.

Since we are interested in separability analysis, assertions state whether two quantum bits are entangled or not. Moreover, since separability is uncomputable (it trivially reduces to the halt problem since on can add $\mathfrak{Cnot}(q_i, q_j)$ as a last line of a program in such a way that q_i and q_j are entangled iff the computation stops), assertions are safe approximations: if an assertion state that two quantum bits are separable then they really are, whereas if two quantum bits are stated entangled by an assertion, it is possible that in reality they are not.

4.1 Assertions

Definition 8. *Terms and assertions are defined by the following grammar:*

$$e, e' \quad ::= \quad u \mid q_i \mid \langle e, e' \rangle \mid \pi_i(e)$$

$$\begin{array}{rl} C, C' ::= & u \leftrightarrow v \mid \| e \mid e = e' \\ & \neg C \mid C \vee C' \mid C \wedge C' \mid C \implies C' \mid \forall u.C \mid \exists u.C \\ & \{C\} e_1 \bullet e_2 = e_3 \{C'\} \end{array}$$

Where u, v are names from a countable set of anchor names.

The idea behind assertions is the following: every subterm of a program is identified in assertions by an anchor, which is simply a unique name. The anchor is the logical counterpart of the program. Note that the name of quantum bits are considered as ground terms.

Assertion $u \leftrightarrow v$ means that the quantum bit identified by u is possibly entangled with v. Notice that $\neg u \leftrightarrow v$ means that it is sure that u and v are separable. $\| u$ means that it is for sure that the quantum bit is in a base state (it can be seen as $\alpha|b\rangle$ where b is either $\mathbf{1}$ or $\mathbf{0}$). Thus $\neg \| u$ means that u may not be in a base state (here the approximation works the other around). Assertion $\{C\} e_1 \bullet e_2 = e_3 \{C'\}$ is used to handle higher order functions. It is the evaluation formula. e_3 binds its free occurrences in C'. following [2], C, C' are called internal pre/post conditions. The idea is that invocation of a function denoted by e_1 with argument e_2 under the condition that the initial assertion C is satisfied by the current quantum state evaluates in a new quantum state in which C' is satisfied. C' describes the new entanglement and purity relations.

The other assertions have their standard first order logic meaning. Notice that in $\forall$ and $\exists$ binder are only meant to be used on quantum bits. That is $\forall u.C$ means that u is either of the form q_i or of the form x, with x of type $\mathbf{B}^\circ$ but cannot be of the form $\langle e, e' \rangle$.

In the following we T (resp. F) for the following tautology (resp. antilogy) $u = u$ (resp $\neg(u = u)$).

Definition 9 (Assertion typing).

– *A logical term t is well typed of type τ, written $\Gamma; \Lambda; \Delta \vdash t : \tau$ if it can be derived from the following rules:*

$$\frac{(u : \tau) \in \Gamma; \Lambda; \Delta}{\Gamma; \Lambda; \Delta \vdash u : \tau}[TAsAx]$$

$$\frac{}{\Gamma; \Lambda; \Delta \vdash q_i : \mathbf{B}^\circ}[TAsQ]$$

$$\frac{\Gamma; \Lambda; \Delta \vdash e : \tau \qquad \Gamma; \Lambda; \Delta \vdash e' : \tau'}{\Gamma; \Lambda; \Delta \vdash \langle e, e' \rangle : \tau \otimes \tau'}[TAs\otimes]$$

$$\frac{\Gamma; \Lambda; \Delta \vdash u : \tau_1 \otimes \tau_2}{\Gamma; \Lambda; \Delta \vdash \pi_i(u) : \tau_i}[TAs\pi_i]$$

with $i \in \{1, 2\}$.

– *An assertion C is well typed under context $\Gamma;\Lambda;\Delta$ written $\Gamma;\Lambda;\Delta \vdash C$ if it can be derived from the following rules:*

$$\frac{\Gamma;\Lambda;\Delta \vdash e : \mathbf{B}^\circ \qquad \Gamma;\Lambda;\Delta \vdash e' : \mathbf{B}^\circ}{\Gamma;\Lambda;\Delta \vdash e \leftrightarrow e'}[TAs \leftrightarrow]$$

$$\frac{\Gamma;\Lambda;\Delta \vdash e : \mathbf{B}^\circ}{\Gamma;\Lambda;\Delta \vdash \| e}[TAs\|]$$

$$\frac{\Gamma;\Lambda;\Delta \vdash e : \tau \qquad \Gamma;\Lambda;\Delta \vdash e' : \tau}{\Gamma;\Lambda;\Delta \vdash e = e'}[TAs =]$$

$$\frac{\Gamma;\Lambda;\Delta \vdash C}{\Gamma;\Lambda;\Delta \vdash \neg C}[TAs\neg]$$

$$\frac{\Gamma;\Lambda;\Delta \vdash C \qquad \Gamma;\Lambda;\Delta \vdash C'}{\Gamma;\Lambda;\Delta \vdash C \wedge C'}[TAs\wedge]$$

$$\frac{\Gamma;\Lambda;\Delta \vdash C \qquad \Gamma;\Lambda;\Delta \vdash C'}{\Gamma;\Lambda;\Delta \vdash C \vee C'}[TAs\vee]$$

$$\frac{\Gamma;\Lambda;\Delta \vdash C \qquad \Gamma;\Lambda;\Delta \vdash C'}{\Gamma;\Lambda;\Delta \vdash C \implies C'}[TAs \implies]$$

$$\frac{\Gamma;\Lambda;\Delta, u : \mathbf{B}^\circ \vdash C}{\Gamma;\Lambda;\Delta \vdash \forall u.C}[TAs\forall]$$

$$\frac{\Gamma;\Lambda;\Delta, u : \mathbf{B}^\circ \vdash C}{\Gamma;\Lambda;\Delta \vdash \exists u.C}[TAs\exists]$$

$$\frac{\Gamma;\Lambda,\Lambda';\Delta \vdash C \qquad \Gamma;\Lambda,\Lambda';\Delta, e3 : \tau \vdash C' \qquad \begin{array}{c}\Gamma;\Lambda';\Delta \vdash e2 : \sigma \\ \Gamma;\Lambda;\Delta \vdash e1 : \sigma \to \tau\end{array}}{\Gamma;\Lambda,\Lambda';\Delta \vdash \{C\}e1 \bullet e2 = e3\{C'\}}[TAsEV]$$

Assertion typing rules may be classified in two categories. The first one is the set of rules insuring correct use of names with respect to the type of the term denoted by them. It is done by rules $[TAs \leftrightarrow]$ $[TAs\|]$ $[TAs =]$ $[TAs\forall]$ $[TAs\exists]$ and $[TAsEV]$. The second set of rules is used to structurally check formulas: $[TAs\neg]$ $[TAs\wedge]$ $[TAs\vee]$, and $[TAs \implies]$.

4.2 Semantics

We now formalize the intuitive semantics of assertions. For this, we abstract the set of quantum bits to an abstract quantum state. The approximation (we are conservative in saying that two quantum bits are entangled and in stating the non-purity of a quantum bits) is done at this level. It means that for a given quantum state there are several abstract quantum state acceptable. For instance stating that all quantum bits are entangled, and not one of them is in a base state, which is tautological, holds as an acceptable abstract quantum state for any actual quantum state. The satisfaction of an assertion is done relatively to the abstract operational semantics. We develop an abstract operational semantics in order to abstractly execute λ_L^Q programs.

Abstract quantum state and abstract operational semantics Let the fixed set of n quantum bits be named S in the following of this section. Let also suppose that the quantum state of S is described by $|\varphi\rangle$ a normalized vector of $\mathbb{C}^2$.

Definition 10 (Abstract quantum state). *An abstract quantum state of S (AQS for short) is a tuple $A = (\mathcal{R}, \mathcal{P})$ where $\mathcal{P} \subseteq S$ and $\mathcal{R}$ is a partial equivalence relation on $(S \setminus \mathcal{P}) \times (S \setminus \mathcal{P})$.*

Relation $\mathcal{R}$ is a PER since it describes an approximation of the entanglement relation and there is not much sens in talking about the entanglement of a quantum bit with itself. Indeed because of the no-cloning property it is not possible to have programs $p : \mathbf{B}^\circ \times \mathbf{B}^\circ \to \tau$ requiring two non entangled quantum bits and to type $(p\ \ \langle q_i, q_i\rangle)$.

The equivalence class of a quantum bit q with relation to an abstract quantum state $A = (\mathcal{R}, \mathcal{P})$ is written $\overline{q}^A$.

Definition 11 (AQS and quantum state adequacy). *Let S be described by $|\varphi\rangle$ and $A = (\mathcal{R}, \mathcal{P})$ an AQS of S. A is adequate with regards to $|\varphi\rangle$, written $A \models |\varphi\rangle$, iff for every $x, y \in S$ such that $(x, y) \notin \mathcal{R}$ then x, y are separable w.r.t. $|\varphi\rangle$ and for every $x \in \mathcal{P}$ then the measurement of x is deterministic.*

Suppose that $S = \{q_1, q_2, q_3\}$ and $|\varphi\rangle = 1/\sqrt(2)(|\mathbf{0}\rangle + |\mathbf{1}\rangle \otimes 1/\sqrt(2)(|\mathbf{0}\rangle + |\mathbf{1}\rangle \otimes |\mathbf{1}\rangle$ then:

- $A = (\{(q_1, q_2), (q_2, q_1)\}, \{q_3\})$
- $A' = (\{(q_1, q_2), (q_2, q_1), (q_2, q_3), (q_3, q_2), (q_3, q_1), (q_1, q_3)\}, \emptyset)$

are such that $A \models |\varphi\rangle$ and $A' \models |\varphi\rangle$. On the other hand:

- $B = (\{(q_1, q_2), (q_2, q_1)\}, \{q_2, q_3\})$
- $B' = (\emptyset, \{q_3\})$

are not adequate abstract quantum states with relation to $|\varphi\rangle$.

We now give a new operational semantics of λ_L^Q terms based on abstract quantum states transformation.

Definition 12 (Abstract operational semantics). *We define an abstract operational semantics of a term M such that $\Gamma; \Lambda \vdash M : \tau$ between AQS as :*

$$[A, M] \to_{\mathcal{A}}^{\Gamma,\Lambda} [A', M']$$

We often write $\rightarrow_{\mathcal{A}}$ instead of $\rightarrow_{\mathcal{A}}^{\Gamma,\Lambda}$ when typing contexts play no role or can be inferred from the context.

Reduction rules are the same ones as those of definition 7 for the functional part of the calculus where the quantum state is replaced with an abstract state. We have the following rules for the quantum actions:

$$\frac{}{[(\mathcal{R},\mathcal{P}),(\mathfrak{T}\quad q_i)] \rightarrow_{\mathcal{A}} [(\mathcal{R},\mathcal{P}),q_i]}[PHS_{\mathcal{A}}]$$

$$\frac{}{[(\mathcal{R},\mathcal{P}),(\mathfrak{H}\quad q_i)] \rightarrow_{\mathcal{A}} [(\mathcal{R},\mathcal{P}\setminus\{q_i\}),q_i]}[HDR_{\mathcal{A}}]$$

$$\frac{}{[(\mathcal{R},\mathcal{P}),(\mathsf{meas}\quad q_i)] \rightarrow_{\mathcal{A}} [(\mathcal{R}\setminus q_i,\mathcal{P}\cup\{q_i\}),\frac{\mathbf{1}}{\mathbf{0}}]}[MET_{\mathcal{A}}]$$

$$\frac{}{[(\mathcal{R},\mathcal{P}),(\mathfrak{Cnot}\quad \langle q_i,q_j\rangle)] \rightarrow_{\mathcal{A}} [(\mathcal{R},\mathcal{P}),\langle q_i,q_j\rangle]}[CNO1_{\mathcal{A}}]\textit{if } q_i\in\mathcal{P}$$

$$\frac{}{[(\mathcal{R},\mathcal{P}),(\mathfrak{Cnot}\quad \langle q_i,q_j\rangle)] \rightarrow_{\mathcal{A}} [(\mathcal{R}\cdot q_i\leftrightarrow q_j,\mathcal{P}\setminus\{q_i,q_j\}),\langle q_i,q_j\rangle]}[CNO0_{\mathcal{A}}]\textit{if } q_i\notin\mathcal{P}$$

Where $\frac{\mathbf{1}}{\mathbf{0}}$ is non deterministically $\mathbf{1}$ or $\mathbf{0}$, $\mathcal{R}\backslash q_i$ is the equivalence relation such that if $(x,y)\in relentangle$ and $x\neq q_i$ or exclusive $y\neq q_i$ then $(x,y)\in\mathcal{R}\setminus q_i$ otherwise $(x,y)\notin\mathcal{R}\setminus q_i$, and where $\mathcal{R}\cdot q_i\leftrightarrow q_j$ is the equivalence relation $\mathcal{R}$ in which the equivalence classes of q_i,q_j have been merged together.

Note that this abstract semantics is not deterministic since it non deterministically gives $\mathbf{1}$ or $\mathbf{0}$ as result of a measure. Its correctness can hurt the intuition since the measurement of a quantum bit in a base state, say $|\mathbf{1}\rangle$, can never produce $|\mathbf{0}\rangle$. Note also that since our system is normalizing the number of all possible abstract executions is finite. Hence, computable.

Definition 13 (Abstract program semantics). *Consider an AQS A, the semantics of program $\Gamma;\Lambda\vdash M:\tau$ under A, written $[\![M]\!]_A^{\Gamma;\Lambda}$, is the set of A' such that $[A,M]\rightarrow_{\mathcal{A}}^{*}[A',V]$ where V is a value.*

Notice that the abstract semantics of a program is a collecting semantics. It may explore branches that are never going to be used in actual computation. Indeed in the operational semantics measurement gives a non deterministic answer. Nevertheless, correctness is ensured by the if judgment rules (see rule $[IF_J]$ in definition 20).

Proposition 2. *Let $A\models|\varphi\rangle$, $\Gamma;\Lambda\vdash M:\tau$. Suppose that $[|\varphi\rangle,M]\rightarrow_{\gamma}^{*}[|\varphi'\rangle,V]$ then there exists $A'\models|\varphi'\rangle$ such that $[A,M]\rightarrow_{\mathcal{A}}^{*}[A',V]$.*

Proof. The proof is done by induction on the number of steps of the reduction between $[|\varphi\rangle,M]$ and $[|\varphi'\rangle,V$. The proposition is clearly true if there is 0 step since $M=V$, $\varphi=\varphi'$ and $A'=A$ proves the result.

Now consider the last rule used. If this rule is one of the purely functional part of the calculus (see def. 7) the proposition follow directly from the induction hypothesis since the AQS is not changed. We thus have the following possibilities for the last rule:

- It is $[PHS_{\mathcal{A}}]$: If the qbit q on which phase is applied is a base state it can be written $\alpha|l\rangle$ with l being either $\mathbf{1}$ or $\mathbf{0}$. Thus $\mathfrak{T}q = \exp^{i\pi/4}\alpha$, thus still a base state. Hence $\mathcal{P}$ remains unchanged.
- It is $[HDR_{\mathcal{A}}]$: if $(\mathcal{R},\mathcal{P}) \models \varphi$, then $(\mathcal{R},\mathcal{P} \setminus \{q_i\}) \models (\mathfrak{H}_i \quad |\varphi\rangle)$ because of definition 11 since in $(\mathfrak{H}_i \quad |\varphi\rangle)$, any q_j is in a non base state only if it is in a non base state in $|\varphi\rangle$.
- It is $[MET_{\mathcal{A}}]$: After the measure the qubit vanishes. Moreover concrete measure probabilistically produces $\mathbf{1}$ or $\mathbf{0}$. Regarding the concrete result one can choose the appropriate value as result of the abstract measure, moreover the measured qubit is in a base state (hence the $\mathcal{P} \cup \{q_i\}$).
- It is $[NEW_{\mathcal{A}}]$: then by definition $|\varphi'\rangle = |\mathbf{1}\rangle \otimes |\varphi\rangle$, hence quantum in a base state in φ remain in a base state in φ', moreover the new qubit is in a base state.
- It is $[CNO0_{\mathcal{A}}]$: If the two qubits $q_i = \alpha|l\rangle, q_j = \beta|l'\rangle$ are in a base state then
  - If $l = \mathbf{1}$ then $\mathfrak{Cnot}(\alpha|\mathbf{1}\rangle \otimes \beta|l'\rangle) = \alpha|\mathbf{1}\rangle \otimes \beta|\neg l'\rangle$
  - If $l = \mathbf{0}$ then $\mathfrak{Cnot}(\alpha|\mathbf{0}\rangle \otimes \beta|l'\rangle) = \alpha|\mathbf{0}\rangle \otimes \beta|l'\rangle$

  in both cases we obtain two separable qubits.
  If only $q_i = \alpha'|l\rangle$ is in a base state and $q_j = \alpha|\mathbf{0}\rangle + \beta|\mathbf{1}\rangle$ is not.
  - If $l = \mathbf{1}$ then $\mathfrak{Cnot}(\alpha|\mathbf{1}\rangle \otimes \alpha|\mathbf{0}\rangle + \beta|\mathbf{1}\rangle) = \alpha'|\mathbf{1}\rangle \otimes \beta|\mathbf{1}\rangle + \alpha|\mathbf{0}\rangle$
  - If $l = \mathbf{0}$ then $\mathfrak{Cnot}(\alpha'|\mathbf{1}\rangle \otimes \alpha|\mathbf{1}\rangle + \beta|\mathbf{0}\rangle) = \alpha'|\mathbf{1}\rangle \otimes \alpha|\mathbf{1}\rangle + \beta|\mathbf{0}\rangle)$

  here also we obtain two separable qubits. Moreover in all cases q_i remains in a base state.
- It is $[CNO1_{\mathcal{A}}]$: The property follows from induction hypothesis and from the fact that $\mathcal{R}$ and $\mathcal{P}$ are safe approximations.

Semantics of entanglement assertions We now give the semantics of a well typed assertion with relation to a concrete quantum state. It is done via an abstract quantum state which is adequate with regards to the concrete quantum state. The idea is as follows: if $|\varphi\rangle \models A$, and if $\Gamma;\Lambda;\Delta \vdash C$ then we define the satisfaction relation $\mathcal{M}^{\Gamma;\Lambda;\Delta} \models C$, which states that under a proper model depending on the typing context, then C is satisfied. Basically it amounts to check two properties : whether or not two quantum bits are in the same entanglement equivalence class and whether or not a particular quantum bit is in base state.

Definition 14 (Abstract observational equivalence). *Suppose that $\Gamma;\Lambda \vdash M, M' : \tau$. M and M' are observationally equivalent, written $M \equiv_A^{\Gamma,\Lambda} M'$, if and only if for all context $C[.]$ such that $\cdot;\cdot \vdash C[M], C[M'] : \mathbf{B}$ and for all AQS A we have*

$$[\![C[M]]\!]_A^{\Gamma,\Lambda} = [\![C[M']]\!]_A^{\Gamma,\Lambda}$$

The equivalence class of M is denoted by $\widetilde{M}_A^{\Gamma,\Lambda}$, by extension we say that the type of this equivalence class is τ.

Definition 15 (Abstract values). *In assertion typing context $\Gamma;\Lambda;\Delta$, an abstract value $v_{A,tau}^{\Gamma;\Lambda;\Delta}$ of type τ, where $\tau \neq \sigma \otimes \sigma'$, with relation to context $\Gamma;\Lambda;\Delta$ and AQS $A = (\mathcal{R}, \mathcal{P})$ is:*

- *An equivalence class of type τ for $\equiv_A^{\Gamma,\Lambda}$, if $\tau \neq \mathbf{B}^\circ$.*
- *a pair (C, b) formed by an equivalence class C of $\mathcal{R}$ and a boolean b (the idea being that if b is true then the denoted qubit is in $\mathcal{P}$).*

If $\tau = \sigma' \otimes \sigma''$, then $v_{A,\tau}^{\Gamma;\Lambda;\Delta}$ is a pair (v', v'') formed by abstract values of respective types σ', σ''.

The set of abstract values under an AQS A, typing context $\Gamma;\Lambda;\Delta$ and for a type τ is written $\Xi_{A,\tau}^{\Gamma;\Lambda;\Delta}$.

Abstract values are used to define the interpretation of free variables. Since in a Given an assertion typing context $\Gamma;\Lambda;\Delta$ more than one type may occur we need to consider collections of abstract values of the different types that occur in $\Gamma;\Lambda;\Delta$: we write $\Xi_{\Gamma;\Lambda;\Delta}$ the disjoint union of all $\Xi_\tau^{\Gamma;\Lambda;\Delta}$ for every τ in $\Gamma;\Lambda;\Delta$.

Definition 16 (Models). *A $\Gamma;\Lambda;\Delta$ model is a tuple $\mathcal{M}^{\Gamma;\Lambda;\Delta} = \langle A, \mathcal{I} \rangle$, where A is an AQS, $\mathcal{I}$ is a map from variables defined in $\Gamma;\Lambda;\Delta$ to $\Xi_{\Gamma;\Lambda;\Delta}$.*

In order to deal with evaluation and quantified formulas we need to define a notion of model extension.

Definition 17 (Model extensions). *Let $\mathcal{M}^{\Gamma;\Lambda;\Delta} = \langle A, \mathcal{I} \rangle$ be a model, then the model $\mathcal{M}'$ written $\mathcal{M} \cdot x : v = \langle A, \mathcal{I}' \rangle$, where $v \in \Xi_{A,\tau}^{\Gamma;\Lambda;\Delta}$ is defined as follows:*

- *the typing context of $\mathcal{M}'$ is $\Gamma;\Lambda;\Delta, x : \tau$.*
- *If the type of x is $\tau = \sigma \otimes \sigma'$, then v is a couple made of abstract values V', v'' of respective type σ, σ'.*
- *If the type of x is $\mathbf{B}^\circ$: if $v = (C, \mathbf{1})$ then $A' = (\mathcal{R} \cup C, \mathcal{P}' \cup \{x\})$, otherwise if $v = (C, \mathbf{0})$ then $A' = (\mathcal{R} \cup C, \mathcal{P}')$.*
- *If the type of x is $\sigma \neq \mathbf{B}^\circ$, then: $\mathcal{I}'(y) = \mathcal{I}(y)$ for all $x \neq y$ and $\mathcal{I}'(x) = v$*

We now define term interpretation. It is standard and amounts to an interpretation of names into abstract values of the right type.

Definition 18 (Term interpretation). *Let $\mathcal{M}^{\Gamma,\Lambda} = \langle A, \mathcal{I}, \tau \rangle$ be a model, the interpretation of a term u is defined by:*

- $[\![u]\!]_{\mathcal{M}} = \mathcal{I}(u)$ *if the type of u is not $\mathbf{B}^\circ$.*
- $[\![q_i]\!]_{\mathcal{M}} = (\overline{q_i}^A, b_i^A)$*, where b_i^A is true iff $q_i\mathcal{P}$ with $A = \langle \mathcal{R}, \mathcal{P} \rangle$.*
- $[\![\langle e, e' \rangle]\!]_{\mathcal{M}} = \langle [\![\mathcal{M}]\!]_A, [\![e']\!]_{\mathcal{M}} \rangle$

Definition 19 (Satisfaction). *The satisfaction of an assertion C in the model $\mathcal{M} = \langle A, \mathcal{I} \rangle$, is written $\mathcal{M} \models C$, is inductively defined by the following rules:*

- $\mathcal{M} \models u \leftrightarrow v$ *if* $(\pi_1([\![u]\!]_{\mathcal{M}}), \pi_1([\![v]\!]_{model})) \in \mathcal{R}_A$.
- $\mathcal{M} \models \| u$ *if* $\pi_2([\![u]\!]_{\mathcal{M}})$ *is true.*

- $\mathcal{M} \models e_1 = e_2$ *if* $[\![e_2]\!]_A = [\![e_1]\!]_A$.
- $\mathcal{M} \models \neg C$ *if* $\models$ *does not satisfy* C.
- $\mathcal{M} \models C \vee C'$ *if* $\mathcal{M} \models C$ *or* $\mathcal{M} \models C'$.
- $\mathcal{M} \models C \wedge C'$ *if* $\mathcal{M} \models C$ *and* $\mathcal{M} \models C'$.
- $\mathcal{M} \models C \implies C'$ *if* $\mathcal{M} \models C$ *implies* $\mathcal{M} \models C'$.
- $\mathcal{M} \models \forall u.C$ *if for all model* $\mathcal{M}' = \mathcal{M} \cdot u.v$*, one has* $\mathcal{M}' \models C$.
- $\mathcal{M} \models \exists u.C$ *if there is an abstract value* v *such that if* $\mathcal{M}' = \mathcal{M} \cdot u.v$*, one has* $\mathcal{M}' \models C$.
- $\mathcal{M} \models \{C\}e_1 \bullet e_2 = e_3\{C'\}$ *if for all models* ${\mathcal{M}'}^{\Gamma;\Lambda;\Delta} = \langle A', \mathcal{I}' \rangle$ *such that* ${\mathcal{M}'}^{\Gamma;\Lambda;\Delta'} \models C$*, with the following conditions:* $\Gamma;\Lambda;\Delta \vdash e_1 : \sigma \to \tau$*, and* $\Gamma;\Lambda;\Delta \vdash e_2 : \sigma$ *such that for all terms* $t_1 \in [\![e_1]\!]_{\mathcal{M}'}, t_2 \in [\![e_2]\!]_{\mathcal{M}'}$ *one has*

  - $[A, (t_1 \quad t_2)] \to^*_{\mathcal{A}} [A', V]$
  - *we have two sub-cases:*

    1. τ *is* $\mathbf{B}^\circ$ *and* $V = q_i$ *and* $\mathcal{M}' = \mathcal{M} \cdot e_3 : (\overline{q_i}^{A'}, q_i \in \mathcal{P}_{A'})$
    2. τ *is not* $\mathbf{B}^\circ$ *and* $\mathcal{M}' \cdot e_3 : \widetilde{V}_A^{\Gamma;\Lambda;\Delta,\tau} \models C'$

4.3 Judgments and proof rules

We now give rules to derive judgments of the form $\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} u\{C'\}$. Those judgments bind u in C', thus u cannot occur freely in C. There are two kinds of rules: the first one follow the structure of M, the second one are purely logical rules.

Definition 20 (Language rules). *Let* $\Gamma;\Lambda \vdash M : \tau$*, we define the judgment* $\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} u\{C'\}$ *inductively as follows:*

$$\frac{\{C \wedge \|u\}N :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ\otimes\mathbf{B}^\circ} \langle u,v\rangle\{C'\}}{\{C \wedge \|u\}(\mathfrak{Cnot}\ N) :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ\otimes\mathbf{B}^\circ} \langle u,v\rangle\{C'\}}[CNOT1_J]$$

$$\frac{\{C\}N :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ\otimes\mathbf{B}^\circ} \langle u,v\rangle\{C'\}}{\{C\}(\mathfrak{Cnot}\ N) :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ\otimes\mathbf{B}^\circ} \langle u,v\rangle\{C' \wedge u \leftrightarrow v\}}[CNOT2_J]$$

$$\frac{\{C\}N :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ} v\{C'\}}{\{C\}(\mathfrak{H}\ N) :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ} v\{C'[\neg\|v]\}}[HAD_J]$$

$$\frac{\{C\}N :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ} u\{C'\}}{\{C\}(\mathfrak{T}\ N) :^{\Gamma;\Lambda;\Delta,\mathbf{B}^\circ} u\{C'\}}[PHASE_J]$$

$$\frac{}{\{C[u/x]\}x :^{\Gamma;\Lambda;\Delta,u:\tau;\tau} u\{C\}}[VAR_J]$$

$$\frac{c \in \{\mathbf{1},\mathbf{0}\}}{\{C\}c :^{\Gamma;\Lambda;\Delta,u:\mathbf{B};\mathbf{B}} u\{C\}}[CONST_J]$$

$$\frac{\{C\}M :^{\Gamma;\Lambda;\Delta} \Gamma;\Lambda;\Delta;\mathbf{B}^\circ\{u\}C'}{\{C\}\mathsf{meas}\ M :^{\Gamma;\Lambda;\Delta,v:\mathbf{B};\mathbf{B}} v\{C'[-u] \wedge \|u\}}[MEAS_J]$$

$$\frac{\{C\}M :^{\Gamma;\Lambda;\Delta;\mathbf{B}} b\{C_0\} \quad \{C_0[\mathbf{1}/b]\}N :^{\Gamma;\Lambda;\Delta;\tau} x\{C'\} \quad \{C_0[\mathbf{0}/b]\}P :^{\Gamma;\Lambda;\Delta;\tau} x\{C'\}}{\{C\}\mathsf{if}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P :^{\Gamma;\Lambda;\Delta,u:\tau;\tau} u\{C'\}}[IF_J]$$

$$\frac{\{C\}M :^{\Gamma;\Lambda;\Delta;\sigma\to\tau} m\{C_0\} \quad \{C_0\}N :^{\Gamma;\Lambda;\Delta;\sigma} n\{C_1 \wedge \{C_1\}m \bullet n = u\{C'\}\}}{\{C\}(M \quad N) :^{\Gamma;\Lambda;\Delta,u:\tau;\tau} u\{C'\}}[APP_J]$$

$$\frac{\{C^{-x} \wedge C_0\}M :^{\Gamma;\Lambda;\Delta;\tau} m\{C'\}}{\{C\}\lambda x : M. :^{\Gamma[-x];\Lambda[-x];\Delta,u:\sigma\to\tau;\sigma\to\tau} u\{\forall x.\{C_0\}u \bullet x = m\{C'\}\}}[ABS_J]$$

$$\frac{\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} m\{C_0\} \quad \{C_0\}N :^{\Gamma;\Lambda;\Delta;\sigma} n\{C'[m/u,n/v]\}}{\{C\}\langle M,N\rangle :^{\Gamma;\Lambda;\Delta,u:\tau,v:\sigma;\tau} \langle u,v\rangle\{C'\}\}}[\times_J]$$

$$\frac{\{C\}M :^{\Gamma;\Lambda;\Delta;\tau_1\otimes\tau_2} m\{C'[\pi_i(m)/u]\} \qquad i \in \{1,2\}}{\{C\}\pi_i M :^{\Gamma;\Lambda;\Delta u:\tau_i;\tau_i} u\{C'\}}[\pi_J]i$$

Where in rule $[HAD_J]$, if there exists C'' such that $C'' \wedge \|u \equiv C'$ the assertion $C'[\neg\|v]$ is $C'' \wedge \neg\|u$ otherwise it is $C'\neg\|u$. In $[MEAS_J]$, the assertion $C'[-u]$ is C' where all assertions containing u have been deleted. In $[ABS_J]$, C^{-x} means that x does not occur freely in C. In $[VAR_J]$, $C[u/x]$ is the assertion C where all free occurrences of x have been replaced by u.

Judgment of the purely functional fragment are standard see [2]. We have just modified the way to handle couples in order to ease manipulations, but we could have used projections instead of introducing two different names. Regarding the quantum fragment, rule $[CNOT1_J]$ has no influences over quantum entanglement since the first argument of the $\mathfrak{Cnot}$ is in a base state; rule $[CNOT2_J]$ introduces an entanglement between the two arguments of the $\mathfrak{Cnot}$ operator. Notice that it is not useful to introduce all entanglement pairs introduced. Indeed, since the entanglement relation is an equivalence relation one can safely add to judgment (see logical rules that follow in def. 21) statements for transitivity, reflexivity and symmetry of entanglement relation, for instance $\forall x, y, z.x \leftrightarrow y \wedge y \leftrightarrow z \implies x \leftrightarrow z$ for transitivity. Indeed any abstract quantum state, by definition, validates those statements which will be implicitly supposed in the following. As we saw in the proof of proposition 2, the phase gate does not change the fact that a quantum bit is in a base state, whereas the Hadamard gate may make him not in a base state, hence explaining the conclusions of rules $[HAD_J]$ $[PHASE_J]$.

We now give purely logical rules. One may see them as an adapted version of standard first order logic sequent calculus.

Definition 21 (Logical rules).

$$\frac{\{C_0\}V : u\{C_0'\} \quad C \vdash C_0' \quad C_0 \vdash C'}{\{C\}V : u\{C'\}}[LOG_J]$$

$$\frac{\{C\}V : u\{C'\}}{\{C \wedge C_0\}V : u\{C' \wedge C_0\}}[promote]$$

$$\frac{\{C \wedge C_0\}V : u\{C'\}}{\{C\}V : u\{C_0 \implies C'\}}[\implies ELim]$$

$$\frac{\{C\}M : u\{C_0 \implies C'\}}{\{C \wedge C_0\}V : u\{C'\}}[\wedge Elim]$$

$$\frac{\{C_1\}M : u\{C\} \quad \{C_2\}M : u\{C\}}{\{C_1 \vee C_2\}M : u\{C\}}[\vee L]$$

$$\frac{\{C\}M : u\{C_1\} \quad \{C\}M : u\{C_2\}}{\{C\}M : u\{C_1 \wedge C_2\}}[\wedge R]$$

$$\frac{\{C\}M : u\{C'^{-x}\}}{\{\exists x.C\}M : u\{C'\}}[\exists L]$$

$$\frac{\{C^{-x}\}M : u\{C'\}}{\{C\}M : u\{\forall x.C'\}}[\forall R]$$

where $C \vdash C'$ is the standard first order logic proof derivation (see e.g. [8]).

We now give the soundness result relating

Theorem 1 (Soundness). *Suppose that* $\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} u\{C'\}$ *is provable. Then for all model* $\mathcal{M} = \langle A, \mathcal{I}\rangle$*, abstract quantum state* A'*, abstract value* v *such that*

1. $\mathcal{M} \models C$
2. $[A, M] \rightarrow^*_{\mathcal{A}} [A', V]$
3. $v \in \Xi^{\Gamma;\Lambda;\Delta}_{A',\tau}$

then $\mathcal{M} \cdot u : v \models C'$.

Proof. The proof is done by induction on judgment rules. The last judgment rule used can be either a logical or a language one. If it is a logical one, soundness follows from the soundness of first order logic. Observe that we have a value in the promotion rule [*promote*] thus no reductions are possible and the soundness is vacuously valid.

If he last judgment rules used is a language rule, we only consider the quantum fragment (indeed for the functional fragment, the proof follows directly from [2]), thus we have the following cases:

– $[CNOT1_J]$, thus $\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} u\{C'\}$ is in facts $\{C_1 \wedge \|u'\}(\mathfrak{Cnot}\, N) :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ\otimes\mathbf{B}^\circ} \langle u', v'\rangle\{C'\}$. By induction hypothesis we know that if $\mathcal{M} \models C_1 \wedge \|u'$, if $[A, N] \rightarrow^*_{\mathcal{A}} [A', V]$, and $v \in \Xi^{\Gamma;\Lambda;\Delta}_{A',\tau}$, then $\mathcal{M} \cdot \langle u', v'\rangle : v\mathcal{M}C'$. We know that V is a couple of qbits (since judgment is well typed), say $\langle q_i, q_j\rangle$. Now $[A', (\mathfrak{Cnot}\ \langle q_i, q_j\rangle)] \rightarrow_{\mathcal{A}} [A, \langle q_i, q_j\rangle]$ thanks to rule $[CNO1_{\mathcal{A}}]$ and due to the fact that $\mathcal{M} \models \|u'$.
– $[CNOT2_J]$, thus $\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} u\{C'\}$ is in facts $\{C\}(\mathfrak{Cnot}\, N) :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ\otimes\mathbf{B}^\circ} \langle u', v'\rangle\{C' \wedge u' \leftrightarrow v'\}$ we reason similarly as in previous case with the difference that the last abstract operational rule used is $[CNO0_{\mathcal{A}}]$.
– $[HAD_J]$, thus $\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} u\{C'\}$ is in facts $\{C\}(\mathfrak{H}\, N) :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ} u\{C'[\neg\|u]\}$. By induction hypothesis we know that if $\mathcal{M} \models C$, if $[A, N] \rightarrow^*_{\mathcal{A}} [A', V]$, and $v \in \Xi^{\Gamma;\Lambda;\Delta}_{A',\tau}$, then $\mathcal{M} \cdot \langle u\rangle : v\mathcal{M}C'$. Now because judgment is well typed τ is $\mathbf{B}^\circ$, and V is q_i. Thus $[A, (\mathfrak{H}\ q_i)] \rightarrow_{\mathcal{A}} [(\mathcal{R}, \mathcal{P} \setminus \{q_i\}), q_i]$, and clearly $\mathcal{M} \cdot \langle u\rangle : v \models \neg\|u$, the rest is done by induction hypothesis.
– $[PHASE_J]$, thus $\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} u\{C'\}$ is direct by induction hypothesis and considering abstract reduction rule $[PHS_{\mathcal{A}}]$.
– $JDGMEAS$, thus $\{C\}M :^{\Gamma;\Lambda;\Delta;\tau} u\{C'\}$ is in facts $\{C\}(\mathsf{meas}\ N) :^{\Gamma;\Lambda;\Delta;\mathbf{B}^\circ} u\{C'[-u] \wedge \|u]\}$. By induction hypothesis we know that if $\mathcal{M} \models C$, if $[A, N] \rightarrow^*_{\mathcal{A}} [A', V]$, and $v \in \Xi^{\Gamma;\Lambda;\Delta}_{A',\tau}$, then $\mathcal{M} \cdot u : v\mathcal{M}C'$. Now because judgment is well typed τ is $\mathbf{B}^\circ$, and V is q_i. Thus $[A, (\mathsf{meas}\ q_i)] \rightarrow_{\mathcal{A}} [(\mathcal{R}, \mathcal{P} \cup \{q_i\} \setminus \{q_i\}), \frac{\mathbf{1}}{\mathbf{0}}]$, and clearly $\mathcal{M} \cdot u : v \models \|u$, the rest is done by induction hypothesis.

Example 1. The idea of this example is to show how the entanglement logic may be used to analyze non local and non compositional behavior. Suppose 4 qubits, x, y, z, t such that x, t are entangled and y, z are entangled and $\{x, t\}$ separable from $\{y, z\}$. Now if we perform a control not on x, y, then as a side effect z, t are entangled too, even if quantum bits x, y are discarded by measurement. Thus we want to prove the following statement:

$$\{\mathsf{T}\}P : u\{\forall x, y, z, t.\{x \leftrightarrow y \wedge z \leftrightarrow t\}u \bullet y, z = v\{x \leftrightarrow t\}\}$$

where P is the following program

$$\lambda y, z : \mathsf{let}\ \langle u, v\rangle = (\mathfrak{Cnot}\ \langle y, z\rangle)\ \mathsf{in}\ \langle(\mathsf{meas}\ u), (\mathsf{meas}\ v)\rangle.$$

Then using rule $[APP_J]$ we can derive the following judgment on actual quantum bits:

$$\{C\}(P\ \langle q_2, q_3\rangle) : \langle u, v\rangle\{q_1 \leftrightarrow q_4\}$$

where C denotes the following assertion : $q_1 \leftrightarrow q_2 \wedge q_3 \leftrightarrow q_4$. This judgment is remarkable in the fact that it asserts on entanglement properties of q_1, q_4 while those two quantum bits do not occur in the piece of code analyzed.

5 Conclusion

In this paper we have proposed a logic for the static analysis of entanglement for a functional quantum programing language. We have proved that this logic is safe and sound: if two quantum bits are provably separable then they are not entangled while if they are provably entangled they could actually be separable. The functional language considered includes higher-order functions. It is, to our knowledge the first proposal to do so and strictly improves over [5] on this respect. We have shown that non local behavior can be handled by this logic.

Completeness of our logic remains an open issue worth of future investigations. We also hope that this setting will allow reasoning examples on quantum algorithms, and that it will provide a useful help for quantum algorithms research in providing a high-level, compositional reasoning tool.

References

1. T. Altenkirch and J. Grattage. A functional quantum programming language. In *20th IEEE Symposium on Logic in Computer Science (LICS 2005)*, pages 249–258. IEEE Computer Society, 2005.
2. M. Berger, K. Honda, and N. Yoshida. A logical analysis of aliasing in imperative higher-order functions. In O. Danvy and B. C. Pierce, editors, *Proceedings of the 10th ACM SIGPLAN International Conference on Functional Programming, ICFP 2005*, pages 280–293, 2005.
3. T. Hoare. An axiomatic basis of computer programming. *CACM*, 12(10):576–580, 1969.
4. M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
5. S. Perdrix. Quantum patterns and types for entanglement and separability. *Electronic Notes Theoretical Computer Science*, 170:125–138, 2007.

6. F. Prost. Taming non-compositionality using new binders. In *Proceedings of Unconventional Computation 2007 (UC'07)*, volume 4618 of *Lecture Notes in Computer Science*. Springer, 2007.
7. P. Selinger and B. Valiron. A lambda calculus for quantum computation with classical control. In P. Urzyczyn, editor, *Typed Lambda Calculi and Applications, 7th International Conference, (TLCA 2005), LNCS 3461*, pages 354–368, Nara, Japan, 2005. Springer.
8. R. M. Smullyan. *First-Order Logic*. Springer, 1968.
9. A. van Tonder. A lambda calculus for quantum computation. *SIAM Journal on Computing*, 33(5):1109–1135, 2004.
10. G. Vidal. Efficient classical simulation of slightly entangled quantum computations. *Physical Review Letters*, 91(14), 2003.